Além do paradigma orientado a objetos, temos também outras formas de lidar e organizar os programas. Uma forma muito comum de categorizar esses paradigmas é dividi-los entre paradigmas imperativos ou declarativos.
Os paradigmas imperativos são aqueles que usam *afirmações* para alterar o estado de um programa, da mesma forma como o modo verbal imperativo no português expressa um comando ou ordem para ser executada. Essa categoria se preocupa com o "como" uma tarefa vai ser executada, o seu passo-a-passo e a sequência dessas etapas. Alguns dos paradigmas que se encaixam aqui são os seguintes:

- Estrutural
- Procedural
- Orientado a objetos

Um exemplo que mostra o paradigma imperativo é a implementação da seguinte função que recebe um vetor e retorna outro vetor com cada um dos valores dobrado:

```
function dobra(vetor){
    let resultados = [];
    for (let i = 0; i < vetor.length ; i++){
        resultados.push(arr[i] * 2);
}
return resultados;
}
```

Podemos notar que passamos as instruções de como percorrer o vetor, qual operação fazer e o que devemos adicionar ao resultado.

Uma outra categoria de paradigma é o *declarativo*. Podemos dizer que uma característica dele é expressar a lógica de um processo sem descrever o seu controle de fluxo. Ou seja, é associado ao "o quê" uma tarefa vai resultar ou retornar. Um paradigma que pode se encaixar nessa categoria é o paradigma funcional.

Uma implementação declarativa do mesmo problema de dobrar os valores de um vetor pode ser feita da seguinte forma:

```
function dobra(vetor){
return vetor.map((item) => item * 2);
}
```

Podemos observar que não foi necessário explicitar como iterar sobre o laço de repetição ou atribuir os novos valores.

No cotidiano temos diversos outros exemplos de afirmações que podem ser consideradas declarativas, como arquivos HTML:

```
<h1> Programação Declarativa</h1>
<p> Estou declarando como quero que o texto apareça, e não dizendo para o computador como renderizar um texto</p>
```

Ou até mesmo as *queries* SQL, nas quais apenas dizemos qual resultado esperamos, sem especificar como a busca deve ser feita:

```
SELECT * FROM Alunos WHERE Escola='Alura';
```

O JavaScript e algumas outras linguagens podem utilizar mais de um paradigma. É comum ouvir o termo "multiparadigma" quando nos referimos a esse tipo de linguagem, e isso traz alguns benefícios, pois permite perfis diferentes de desenvolvedores e sistemas utilizarem uma linguagem em comum.

Claro que um paradigma não é necessariamente melhor que o outro, mas dependendo das circunstâncias podemos utilizar um que seja mais otimizado para determinada aplicação. Algumas funcionalidades precisam alterar o estado de uma aplicação, não podendo ser escritas de forma declarativa, assim como os códigos declarativos que utilizamos podem ter uma implementação imperativa por baixo dos panos.

Você pode conferir mais conteúdos sobre o tema na plataforma da Alura:

- POO: o que é programação orientada a objetos?
- Alura+: O que é Programação Funcional?
- Programação funcional: O que é?
- Linguagens Funcionais – Hipsters #91
- Programação Funcional (e (clojure)) – Hipsters #158